Akaze

Find an image inside another image.

```
(ns wandering-moss
  (:require
    [opencv3.core :refer :all]
    [opencv3.utils :as u])
  (:import
    [org.opencv.features2d Features2d DescriptorExtractor DescriptorMatcher
FeatureDetector]))

(def detector (FeatureDetector/create FeatureDetector/AKAZE))
(def extractor  (DescriptorExtractor/create DescriptorExtractor/AKAZE))
```

```
#'wandering-moss/extractor
```

```
(def original (-> "resources/images/cat.jpg" imread (u/resize-by 0.3)))
(def mat1 (clone original))
(def points1 (new-matofkeypoint))
(.detect detector mat1 points1)

(def show-keypoints1 (new-mat))
(Features2d/drawKeypoints mat1 points1 show-keypoints1 (new-scalar 255 0 0) 0)
(u/mat-view show-keypoints1)
```

```
(def desc1 (new-mat))
(.compute extractor mat1 points1 desc1)
(u/mat-view desc1)
```

```
(def mat2 (-> "resources/images/cat_face.jpg" imread (u/resize-by 0.3)))
(def points2 (new-matofkeypoint))
(.detect detector mat2 points2)

(def show-keypoints2 (new-mat))
(Features2d/drawKeypoints mat2 points2 show-keypoints2 (new-scalar 255 0 0) 0)
(u/mat-view show-keypoints2)

(def desc2 (new-mat))
(.compute extractor mat2 points2 desc2)
```

```
nil
```

```
(def matcher
  (DescriptorMatcher/create DescriptorMatcher/BRUTEFORCE_HAMMINGLUT))
(def matches
  (new-matofdmatch))
(.match matcher desc1 desc2 matches)

(defn best-n-dmatches2[dmatches]
  (new-matofdmatch
    (into-array org.opencv.core.DMatch
      (filter #(< (.-distance %) 10) (.toArray dmatches)))))

(defn draw-matches [_mat1 _points1 _mat2 _points2 _matches draw-method]
  (let[ _mat (new-mat (* 2 (.rows _mat1)) (* 2 (.cols _mat1)) (.type _mat1))
        _sorted-matches (best-n-dmatches2 _matches)]

    (Features2d/drawMatches
              _mat1
              _points1
              _mat2
              _points2
```

```
               _sorted-matches
               _mat
               (new-scalar 255 0 0)
               (new-scalar 0 0 255)
               (new-matofbyte)
               draw-method)
      _mat))

(defn is-a-match [dmatches]
  (> (count (filter #(< (.-distance %) 10) (.toArray dmatches))) 0))

; (is-a-match matches)

(u/mat-view
  (draw-matches mat1 points1 mat2 points2 matches Features2d/NOT_DRAW_SINGLE_POINTS))
```